UNITED STATES PATENT APPLICATION

FOR

METHOD OF DETERMINING THE SIMILARITY OF TWO STRINGS

Inventor:

SENTHIL, Muthu

Assignee:

Oracle International Corporation

Prepared by:
WAGNER, MURABITO & HAO, LLP
Two North Market Street
Third Floor
San Jose, California 95113

ORCL-2003-032-01

METHOD OF DETERMINING THE SIMILARITY OF TWO STRINGS

FIELD OF THE INVENTION

Embodiments of the present invention relate to data processing, and more particularly to determining a similarity between two strings.

BACKGROUND OF THE INVENTION

Legacy computer implemented data processing systems and methods generally required strict matching of equivalent data values. For example, an accounting application includes a database wherein account receivables are logged. Each record may comprise a plurality of fields, such as the name of the enterprise, the date billed, the amount due, the date paid and the amount of payment. A given payment may be received from Widget Inc. in the amount of $500. A given record in the accounting system may contain a record for Acme Widget Co. with an amount due of $723. In early legacy data processing systems and methods, if the data did not exactly match, entry of payment could not be automated. Accordingly an individual would be needed to match the data associated with the received payment to the appropriate record in the accounting system, and thereafter update the record.

As methods of data processing progress, it is desired that the systems and methods are more tolerant of variations between equivalent data values. Accordingly, it is desired that data processing methods and systems are capable of determining a similarity between two strings (e.g., fuzzy string matching). One method of determining a similarity between two strings comprises the Levenshtein distance (LEV) heuristic. The Levenshtein heuristic produces a matrix of hamming distances, which provides a measure of the similarity of the two strings. Another method of determining a similarity between two strings comprises the largest common substring (LCS) heuristic. Accordingly, recent data processing systems and methods, which utilize such a heuristic, can provide some ability to match strings. For example, the payment received from Widget Inc. may be matched to the accounts receivable record for Acme Widget Co. Therefore, some automation of data entry, processing and reporting can be achieved in conventional art data processing systems and methods.

Given two strings of length M and N, respectively, the Levenshtein heuristic is calculated in M times N (MxN) calculations. Similarly, the largest common substring heuristic is calculated in M time N (MxN) calculations. Accordingly, legacy string matching heuristics incur significant processing costs. Thus, string matching heuristics which provide increased string matching capabilities while minimizing computational costs are sought in the data processing arts.

SUMMARY OF THE INVENTION

Embodiments of the present invention provide a method of determining a similarity of a two string. The method comprises calculating a Levenshtein matrix of a first string and a second string. A Levenshtein distance is determined from the Levenshtein matrix. A largest common substring is also determined from the Levenshtein matrix.

Embodiments of the present invention may further comprise calculating a Levenshtein score from the Levenshtein distance and the length of the two strings, and calculating largest common substring score from the length of the largest common substring and the length of the two strings. In addition, the method may also comprise calculating an acronym score. The acronym score may be utilized to reduce false positive determinations, by the Levenshtein heuristic and/or the largest common substring heuristic, that two given strings are similar.

Embodiments of the present invention may further comprise determining one or more numerical scores as a function of the Levenshtein distance and the largest common substring. A first numeric score may be calculated as a sum of a weighted Levenshtein score and a weighted largest common substring score. A second numeric score may be calculated as a sum of a weighted acronym score and the first numeric score. Embodiments of the present invention may further comprise utilizing the first numeric score for determining the similarity between the two strings, when both strings are numeric-type strings. Alternatively, the second numeric score may be utilized for determining the similarity, when one or both strings are character-type strings.

Embodiments of the present invention advantageously combine the Levenshtein distance heuristic and the largest common substring heuristic to reduce the number of computations performed to obtain both heuristics. Accordingly, the total run time of computer implemented embodiments of the present invention is reduced by approximate one half, as compared to calculating the Levenshtein distance heuristic and the largest common substring heuristic separately according to the conventional art.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention is illustrated by way of example and not by way of limitation, in the figures of the accompanying drawings and in which like reference numerals refer to similar elements and in which:

Figure 1 shows a method of determining a similarity between a first string and a second string, in accordance with one embodiment of the invention.

Figures 2A-2B show an exemplary Levenshtein matrix, in accordance with one embodiment of the invention.

Figure 3 shows a method of determining a numerical score representing a similarity between a first string and a second string, in accordance with one embodiment of the invention.

Figures 4A-4H show an exemplary computer implemented method of determining a similarity of a first string and a second string, in accordance with an exemplary embodiment of the invention.

Figure 5 shows an exemplary computing device for implementing embodiments of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

Reference will now be made in detail to the embodiments of the invention, examples of which are illustrated in the accompanying drawings. While the invention will be described in conjunction with these embodiments, it will be understood that they are not intended to limit the invention to these embodiments. On the contrary, the invention is intended to cover alternatives, modifications and equivalents, which may be included within the spirit and scope of the invention as defined by the appended claims. Furthermore, in the following detailed description of the present invention, numerous specific details are set forth in order to provide a thorough understanding of the present invention. However, it is understood that the present invention may be practiced without these specific details. In other instances, well-known methods, procedures, components, and circuits have not been described in detail as not to unnecessarily obscure aspects of the present invention.

Referring to Figure 1, a method of determining a similarity between a first string and a second string (e.g., fuzzy string matching), in accordance with one embodiment of the invention, is shown. As depicted in Figure 1, the method begins with calculating a Levenshtein distance, at 110. The Levenshtein distance is determined by calculating a Levenshtein matrix. If the first string (S1) has a length of m and the second string (S2) has a length of n, the elements of the Levenshtein matrix D can be calculated in accordance with Equation 1, as follows:

$$D[i,j] = \text{minimum of } (D[i-1,j] + 1,\ D[i,j-1] + 1, \text{ or } D[i-1,j-1] + \text{cost}) \qquad (1)$$

Where i = 1 to n, j = 1 to m, element [0,j] = j, element [i,0] = i, and element [0,0] = 0. In one implementation, the cost is 0 if S1[i] = S2[j], and 1 if S1[i] ≠ S2[j]. The Levenshtein distance is specified by element D[m,n] As a heuristic, the greater the Levenshtein distance, D[m,n], the greater the difference is between the two strings.

Referring now to Figures 2A-2B, an exemplary Levenshtein matrix, in accordance with one embodiment of the invention, is shown. As depicted in Figure 2A, the exemplary Levenshtein matrix is calculated for a first string "Acme Widget Co" and a second string "Widget Inc". For illustrative purposes the first string 210 is shown on the left side of the Levenshtein distance matrix along with the corresponding values of i 220. The second string 230 is shown on top of the Levenshtein distance matrix along with the corresponding values of j 240. The hamming distances in the Levenshtein matrix are calculated according to Equation 1. Accordingly, the Levenshtein distance, D[14,10], is equal to eight (8) 250.

Referring again to Figure 1, the method further comprises determining a largest common substring from the Leventshtein matrix, at 120. When calculating the hamming distances of the Levenshtein matrix, a common substring does not involve an insertion, deletion, transposition or the like, and therefore no cost is incurred. Thus, the Levenstein matrix indirectly contains the largest common substring. Accordingly, the largest common substring is identified by determining the longest diagonal of equal hamming distances of lowest value.

As depicted in Figure 2B, the exemplary Levenshtein matrix is shown with the largest common substring denoted 260. The largest common substring is identified by longest diagonal of equal hamming distances of lowest value.

Referring now to Figure 3, a method of determining a numerical score representing a similarity between a first string and a second string, in accordance with one embodiment of the invention, is shown. As depicted in Figure 3, the method begins with calculating a Levenshtein score, S_{LEV}, for the first string and the second string, at 310. The Levenshtein score, S_{LEV}, is calculated in accordance with Equation 2, as follows:

$$S_{LEV} = 1 - ((2 * D[m,n]) / (m + n)) \tag{2}$$

Wherein the Levenshtein score, S_{LEV}, comprises one minus, a product of two (2) and the Levenshtein distance, D[m,n], divided by the sum of the string lengths, (m + n). In one implementation, the Levenshtein distance is determined by calculating a Levenshtein matrix of the first and second strings, as described above. However, any method of determining the Levenshtein distance may also be utilized for calculating the Levenshtein score.

At 315, a weighted Levenshtein score, WS_{LEV}, is calculated in accordance with Equation 3, as follows:

$$WS_{LEV} = W_{LEV} * S_{LEV} \tag{3}$$

Wherein the weighted Levenshtein score, WS_{LEV}, comprises a product of the Levenshtein score (S_{LEV}) and a Levenshtein weight factor, W_{LEV}. In one implementation, the Levenshtein weight factor, W_{LEV}, is substantially between 0.3 and 0.7, with an exemplary value of 0.5.

At 320, a largest common substring score, S_{LCS}, of the first string and the second string is calculated. The largest common substring score, S_{LCS}, is calculated in accordance with Equation 4, as follows:

$$S_{LCS} = (2 * \text{Length of Largest Common Substring}) / (m + n) \qquad (4)$$

Wherein the largest common substring score, S_{LCS}, comprises a product of two (2) and the length of the largest common substring divided by the sum of the string lengths, (m + n). In one implementation, the length of largest common substring is determined from the Levenshtein distance matrix. The longest diagonal of lowest hamming distances in the Levenshtein matrix identifies the largest common substring. However, any method of determining the largest common substring may also be utilized for calculating the length of the largest common substring score.

At 325, a weighted largest common substring score, WS_{LCS}, is calculated in accordance with Equation 6, as follows:

$$WS_{LCS} = W_{LCS} * S_{LCS} \qquad (6)$$

Wherein the weighted largest common substring score, WS_{LCS}, comprises a product of the largest common substring score, S_{LCS}, and a largest common substring weight factor, W_{LCS}. In one implementation, the largest common substring weight factor, W_{LCS}, is substantially between 0.3 and 0.7, with an exemplary value of 0.5. In one implementation, a sum of the Levenshtein weight factor, W_{LEV}, and largest common substring weight factor, W_{LCS}, is set equal to one (1).

In some data processing applications, it may be more likely that character transpositions and omissions have occurred when evaluating the similarity of two strings, such as two strings of numerical data. The largest common substring score is adapted to provide a greater indication of a similarity between two strings when transposition, omissions and the like are typical of the type of strings being compared. Accordingly, the largest common substring weight factor may be increased if the two strings are more likely to contain transpositions, omission and the like. In other data processing applications, it may be more likely that a misspelling or entirely different representation of the data have occurred when evaluating the similarity of two strings, such as two strings of character data. The Levenshtein score is adapted to provide a greater indication of a similarity between two strings in which misspelling or entirely different representations are typical of the type of strings being compared. Accordingly, the Levenshtein weight factor may be increased if the two strings are more likely to contain misspellings or entirely different representations.

At 335, a first numerical score is calculated in accordance with Equation 7, as follows:

$$NS_1 = WS_{LEV} + WS_{LCS} \qquad (7)$$

Wherein the first numerical score, NS_1, comprises the sum of the weighted Levenshtein score, WS_{LEV}, and the weighted largest common substring score, WS_{LCS}.

At 340, an acronym score (S_{ACR}) of the first and second strings is calculated. In one implementation, acronyms of the first and second strings are formed. The first and second acronyms are then compared. If the acronyms are equal than a non-zero score is given, if the acronyms are not equal than a score of zero is given. The acronym score is adapted to rule out a false positive (e.g., GM – General Motors and GE – General Electric) as indicted by the Levenshtein score and/or largest common substring score.

At 345, a weighted acronym score is calculated in accordance with Equation 8, as follows:

$$WS_{ACR} = W_{ACR} * S_{ACR} \qquad (8)$$

Wherein the weighted acronym score, WS_{ACR}, comprises a product of the acronym score, S_{ACR}, and an acronym weight factor, W_{ACR}. In one implementation, the acronym weight factor, W_{ACR}, is substantially between 0.1 and 0.4, with an exemplary value of 0.2.

At 350, a weighted Levenstein/largest common substring score (e.g., weighted first numerical score) is calculated in accordance with Equation 9, as follows:

$$WS_{LEV\text{-}LCS} = W_{LEV\text{-}LCS} * (W_{LEV} * S_{LEV} + W_{LCS} * S_{LCS}) \qquad (9)$$

Wherein the weighted Levenshtein/largest common substring score, $WS_{LEV\text{-}LCS}$, comprises a product of the first numerical score and a Levenshtein/largest common substring weight factor, $W_{LEV\text{-}LCS}$. In one implementation, the Levenshtein/largest common substring weight factor, $W_{LEV\text{-}LCS}$, is substantially between 0.6 and 0.9, with an exemplary value of 0.8.

At 355, a second numerical score is calculated in accordance with Equation 10, as follows:

$$NS_2 = WS_{LEV\text{-}LCS} + WS_{ACR} \qquad (10)$$

Wherein the second numerical score, NS_2, comprises the sum of the weighted Levenshtein/largest common substring score (e.g., first numerical score), $WS_{LEV\text{-}LCS}$, and the weighted acronym score, WS_{ACR}. In one implementation, a sum of the Levenshtein/largest common substring weight factor ($W_{LEV\text{-}LCS}$) and acronym weight factor (W_{ACR}) is equal to one.

The second numerical score is adapted to represent a similarity between a first and second string, which are comprised of characters, or characters and numbers. However, the in the case of a first and second numerical strings, the acronym score is not normally applicable. Therefore, the first numerical score is adapted to represent a similarity between a first and second string, which are comprised of numbers. Accordingly, the method may optionally include determining if the first and second strings are character-type strings (e.g., comprised of characters, or characters and numbers) or numerical-type strings (e.g., comprised of numbers), at 330. If the first and second strings are numerical-type strings, the first numerical score calculated at 335 is utilized. If the first and/or second strings are character-type strings, the second numerical score calculated according to 355 is utilized.

Referring now to Figures 4A-4H, an exemplary computer implemented method of determining a similarity of a first string and a second string, in accordance with an exemplary embodiments of the present invention, is shown. As depicted in Figures 4A and 4B, a method of calculating a Levenshtein distance of a first string and a second string is implemented by the software code. As depicted in Figures 4C and 4D, a method of calculating a largest common substring from a Levenshtein matrix is implemented by the software code. As depicted in Figures 4E and 4F, a method of calculating an acronym score is implemented by the software code. As depicted in Figures 4G and 4H, a method of calculating a consolidated Levenshtein, largest common substring, and acronym score is implemented by the software code.

Referring now to Figure 5, an exemplary computing device for implementing embodiments of the present invention is shown. As depicted in Figure 5, the computing device 510 comprises an address/data bus 520 for communicating information and instructions. One or more processors 530 are coupled with the bus 520 for processing information and instructions. One or more memories 540 are also coupled to the bus 520 for storing information and instructions for the processor(s) 530. The memory 540 may include volatile memory (e.g. random access memory, static RAM, dynamic RAM, and the like), non-volatile memory (e.g. read only memory, programmable ROM, flash memory, EPROM, EEPROM, and the like), mass data storage (e.g. hard disk, optical disk, floppy disk, and the like), and the like.

Optionally, the computing device 510 may further comprise one or more peripheral devices 570 (e.g., mass data storage device, display, keyboard, pointing device, speaker, and the like) coupled to the bus 520. The peripheral devices 570 may provide for inputting and output information and instructions. The computing device 510 may also comprise one or more network interface(s) 550 are also coupled to the bus 520. The network interface 550 provides for communicating with other computing devices across one or more communication channels 560.

Certain elements of the present invention are realized as one or more sequences of instructions (e.g., software code) that reside on a computer-readable medium, such as the memory 540, and are executed by the processor 530. When executed, the information and instructions causes the processor 530 to implement a method of determining a Levenshtein distance for a first and second string utilizing a Levenshtein matrix. The method further comprises determining a largest common substring from the Levenshtein matrix.

When executed, the one or more sequences of instructions may also cause the processor 530 to implement a method of determining a numerical score indicative of the similarity between the first and second strings. In one implementation, the numerical score comprises a consolidated Levenshtein and largest common substring score. In another implementation, the numerical score comprises a consolidated Levenshtein, largest common substring and acronym score.

Accordingly, embodiments of the present invention advantageously combine the Levenshtein distance heuristic and the largest common substring heuristic to reduce the number of computations performed to obtain both heuristics. Thus, reducing the total run time by approximate one half, as compared to calculating the Levenshtein distance heuristic and the largest common substring heuristic separately according to the conventional art. Embodiments of the present invention also advantageously reduce the occurrence of a false positive determination of the similarity between two strings.

The foregoing descriptions of specific embodiments of the present invention have been presented for purposes of illustration and description. They are not intended to be exhaustive or to limit the invention to the precise forms disclosed, and obviously many modifications and variations are possible in light of the above teaching. The embodiments were chosen and described in order to best explain the principles of the invention and its practical application, to thereby enable others skilled in the art to best utilize the invention and various embodiments with various modifications as are suited to the particular use contemplated. It is intended that the scope of the invention be defined by the Claims appended hereto and their equivalents.